A Swarm Grammar-Based Approach to Virtual World Generation

Yasin Raies and Sebastian von Mammen

Julius-Maximilians University, Würzburg, Germany
yasin.raies@stud-mail.uni-wuerzburg.de
sebastian.von.mammen@uni-wuerzburg.de

Abstract. In this work we formulate and propose an extended version of the multi-agent *Swarm Grammar* (SG) model for the generation of virtual worlds. It unfolds a comparatively small database into a complex world featuring terrain, vegetation and bodies of water. This approach allows for adaptivity of generated assets to their environment, unbounded worlds and interactivity in their generation. In order to evaluate the model, we conducted sensitivity analyses at a local interaction scale. In addition, at a global scale, we investigated two virtual environments, discussing notable interactions, recurring configuration patterns, and obstacles in working with SGs. These analyses showed that SGs can create visually interesting virtual worlds, but require further work in ease of use. Lastly we identified which future extensions might shrink required database sizes.

Keywords: Procedural Content Generation · Terrain Generation · Swarm Intelligence · Multi-Agent Systems · Swarm Art

1 Introduction

In computer games, procedural content generation (PCG) techniques are used, e.g., to algorithmically create small assets like weapons [4], large scale terrains [12] or entire planets [6,1]. The latter are examples of *world generation*, yielding vast and diversely populated terrains. Carefully balancing the PCG mechanics can afford playable, diverse and detailed contents at comparatively small workload. Combining the use of sub-generators and pre-generated assets gives the designer a lot of control but also requires a high amount of input data to achieve desirable results. As an alternative, we propose to define a small database of generative agents whose interactions, in turn, unfold into a complex world. This approach bears similarities to the complex interaction networks of enzymes and proteins through interpretation of cellular RNA and DNA that result in complex multicellular development [2]. This approach allows for adaptivity of generated artifacts to their environment through simple rules. For instance, vegetation can sprawl naturally in the presence of water. It also allows for interactivity—the generation can be stopped and resumed, or iteratively influenced. I.e., the user or player immersed into this generated world can remove,

place, and modify actors anywhere at any time [10]. Lastly, the generation can be perpetual, yielding an open-ended evolution of growth and change of a virtual world.

We chose SGs [11], an agent-based extension of L-Systems [8], as the basic model to drive the envisioned agent-based world generation. In SGs, each agent senses its local surroundings and has the means of production of static artifacts as well as differentiation and proliferation into other agents. We adjust and extend the previously introduced SG models [11] into *virtual environment generating Swarm Grammars* (vSGs), focusing on the creation, population, and manipulation of terrains, considering topography, vegetation and bodies of water. We will provide a short introduction of related works in Section 2. Section 3 will formally introduce vSGs. Section 4 will investigate the capabilities of minimal vSG instances based on sensitivity analyses, while Section 5 will focus on the analysis of larger scale structures and link them to the relationships of agents involved in their generation. Section 6 will summarize our findings and conclude with a look on potential future work.

2 Background

In [7], Hendrikx et al. have created a hierarchy of game content, that can be procedurally generated. As can be seen in Figure 1, they distinguish between six classes of content. In this work, we focus on generation of basic building blocks of virtual worlds (*Game Bits*) and their arrangement in a larger scale space (*Game Spaces*). Following the taxonomies in [7] or [20], works that are seminal to our approach are referenced. In particular, these are noise functions for the generation of terrains, L-systems for branched structures, and Boids to model behaviours.

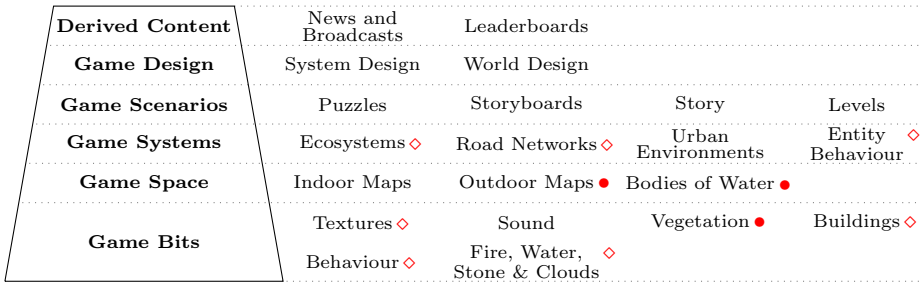

Fig. 1. Hierarchy of game content, as defined by [7]. ●-markers denote classes that we generated through vSGs in this work, while ◇-markers denote classes where we see viable applications of SGs. Figure adapted from [7, p. 4, Figure 1]

.

Sequences of random numbers produced by Pseudo-Random Number Generators (PRNGs) introduce variety in procedurally generated content. As PRNGs

are driven by deterministic functions, feeding them with one and the same seed value ensures reproducibility of the sequences. While random numbers can, e.g., be used to choose, place or scale objects [5], there are methods to structure randomness in multi-dimensional space [13,21]. Such "procedural coherent noise" assigns random values to points in space, for instance to define height-maps of terrains [19]. Creating more cohesive structures, L-Systems, introduced by [8], are parallel string rewrite-systems, often used and initially conceived to describe the growth of plants. They consist of sets of grammatical rules to generate a string of symbols which in turn can be interpreted. There are numerous extensions to basic, context-free L-Systems that solely replace a single character by a string, e.g., context-sensitivity, stochasticity or parameterisation [15]. The established graphical interpretation of L-Systems has been introduced by [14] and parses a string one symbol at a time, informing a drawing agent (a "turtle") to move forward, to rotate or to leap to a previous location. As a result, branching structures as seen in plant morphologies can be retraced.

Unlike the approaches mentioned above, *Birdoids*, *Birdoid Objects* or *Boids* are used to model the behaviour of actors. Introduced in [16], Boids are a method to model the movement behaviour of swarms like flocks of birds or schools of fish. This is achieved through an agent-based approach, where each individual sees their immediate surroundings and moves according to simple urges, like wanting to be close its neighbours or gaining distance to avoid collisions. Due to its efficacy and ease of extendability, the Boid model has received much attention in the artificial life community, but also has been featured in media, like the 1992 Tim Burton film *Batman Returns* [17].

Finally, Swarm Chemistries (SCs) and SGs build on the foundations outlined so far and represent approaches that are rather close to our proposed model. Swarm Chemistries are a Boid-like model where agents are considered chemical reactants, with "spatio-temporal patterns considered as the outcome of chemical reaction[s]" [18]. SCs yield more complex patterns by increasing the number and heterogeneity of deployed agents. SGs, on the other hand, extend L-Systems by interpreting each symbol of the generated string as a Boid agent. As a result, the L-System rules determine the generative behaviour of an SG, whereas the Boid behaviour determines the spatial interaction and interpretation of the generative processes. SGs were created by von Mammen to model dynamic growth, governed by an ecology of interacting agents. An extended form of SGs have been studied in the context of architecture and interactive evolution [11].

3 World-Building Swarm Grammars

To generate virtual worlds, we extended Swarm Grammars with respect to five different aspects: (1) Generalisation of the model, (2) introduction of context-sensitivity in the generative rule sets, (3) terrain-formation capabilities, (4) extension of sensing to create terrain-awareness, (5) introduction of a more expressive energy system.

We subsumed agents and artifacts by the generic concept of an *actor*. Actors are represented by a location, a type-identifier, an energy level, the iteration they have been generated (or age) and their *predecessor*. In order to facilitate adaptivity to the environment not only in movement but also in reproduction, we introduced context-sensitivity in the generative rule sets, thereby defining dependencies between actors within a given neighbourhood range. While von Mammen introduced *extended Swarm Grammars* with similar operational capabilities [9], we incorporated them into the formal model, building on context-sensitive L-Systems. By means of context-sensitivity, markers, so-called stigmergic cues, can be placed in the environment to inform the next steps in a construction process. In order to generate a terrain, following the idea of a height-map, we defined an implicit surface through all current actor positions. We further enabled vSG agents to probe the terrain below them, informing them about its slope and their height. This allowed us to implement new urges: to move along the terrain's downhill-slope, as water droplets would, a self-propulsion tendency as seen in SCs, to prevent stalling, and an axis-constraint, targeting the effective shaping of artifacts. For finer grained control in a world with many different actors, we also added the means to weigh actor types differently on urge calculations. Finally, we diversified the energy system. Previously, SG agents would always inherit the full energy value from their parent and be removed when depleted. In an effort towards finer-grained control of generated structures, vSGs feature new modes of energy distribution/inheritance and an optional final rewrite step on zero energy.

3.1 The vSG Model

We define the starting configuration of an vSG model to be the tuple $SG = (CL, \Delta, \Gamma)$. CL denotes a context-sensitive L-System as described below, $\Delta = \{\Delta_1, \Delta_2, \ldots\}$ a finite set of agent type specifications and $\Gamma = \{\Gamma_1, \Gamma_2, \ldots\}$ a finite set of artifact type specifications. For each agent type Δ_i, parameters of form c_{name,Δ_i} will act as coefficients to movement urges. Parameters e_{name,Δ_i} correspond to energy calculations. a_{max,Δ_i} and v_{name,Δ_i} represent acceleration and velocity parameters, d_{name,Δ_i} and β_{Δ_i} influence the field of view of agents. The artifact types Γ_i only feature a single parameter $\eta_{\Gamma_i,terrain}$, their influence on the terrain. As this parameter is also featured in and between agent types, we have to set all influences of a on b $\eta_{a,b}$, where $a \in \Delta \cup \Gamma$ and $b \in \Delta \cup \{terrain\}$. As vSGs describe an iterative process, we will call $SG(n) = (CL, \Delta, \Gamma)_n = (SG, \mathbb{D}_n, \mathbb{G}_n)$ the n-th iteration of SG, wherein $\mathbb{D}_n = \{d_{1,n}, d_{2,n}, \ldots, d_{k,n}\}$ and $\mathbb{G}_n = \{g_{k+1,n}, g_{k+2,n}, \ldots, g_{l,n}\}$ describe all agents and artifacts present at iteration n. Every element $c_{i,n}$ of the set of all actors in iteration n $\mathbb{D}_n \cup \mathbb{G}_n = \{c_{1,n}, c_{2,n}, \ldots, c_{l,n}\}$ features five properties: Their position $p_{i,n}$, energy $e_{i,n}$, unique identifier $\iota_{i,n}$, back-reference $\rho_{i,n}$ and their type $c^*_{i,n} \in \Delta \cup \Gamma$. Agents additionally feature their velocity $v_{i,n}$ and an individual world center $s_{i,n}$. Finally, the function $h_n(p)$ shall be a representation of the height-map, describing the terrain height below or above $p_{(1\ 0\ 1)}$ after iteration n. With these definitions, we will describe the integration steps of the vSG

model, next: (1) rewrite, (2) movement, and (3) terrain calculations. Note, that iteration n in indices may be omitted for clarity in writing.

Rewriting The L-System $CL = (\alpha, P)$ is at the core of the rewrite step. At iteration $n = 0$, the *axiom* α is converted into $\mathbb{D}_0$ and $\mathbb{G}_0$, while at every second other iteration the productions in P are applied. α may be a word over the alphabet $\Delta \cup \Gamma$. If α were the empty word λ, there would be no actors to simulate. Potential starting positions of actors are not encoded in α. For ease of use, however, implementations might feature a means to set $\mathbb{D}_0$ and $\mathbb{G}_0$ directly. As an implementation default, we place all instantiations of symbols featured in α at $p_{i,0} = \mathbf{0}$, with $e_{i,0} = 10$ energy, a unique id $\iota_{i,0}$ and a velocity vector $v_{i,0} = \mathbf{0}$, if c_i is an agent. The reproduction rules $(p, c, d, \theta, s) \in P$ consist of the *strict predecessor* $p \in \Delta$ and an optional context $c \in (\Delta \cup \Gamma \cup \{\lambda\})^*$ which has to exist within distance d for the rule to be applicable. If c equals the empty word λ, a rule is always applicable. The successor word s may be a word over the alphabet $\Delta \cup \Gamma \cup \{\lambda, \psi\}$, where the optional *persist* symbol ψ may only occur once. ψ will be interpreted as the same agent type as p, but gives the predecessor an identity, allowing it to "live" through a rewrite step and to retain its ι_i. Further, it receives special treatment during energy distribution. In full, productions can be written as $p <_d c \xrightarrow{\theta} s$; if $c = \lambda$ this can be shortened to $p \xrightarrow{\theta} s$. In contrast to SGs, the rule selection probability θ does not represent an absolute probability of rule application. Rather, it describes the frequency of rule selection relative to other currently applicable rules for an agent d_i. This stochastic process may be overridden, if d_i has $e_i \leq 0$ and $e_{zero,d_i^*} \neq ()$. A replacement word akin to the right side of a rule may be provided as s in $e_{zero,i} = (s, v)$, with an additional energy value v assigned to each actor instantiated from s. For each production (p, c, d, θ, s) applied at iteration n, actors corresponding to symbols in s are instantiated. We will refer to the sets of actors as $\mathbb{D}_{n'}$ and $\mathbb{G}_{n'}$. Calling their predecessor $d_{i,n-1}$. All actors $c_{j,n}$ inherit position and velocity, gain a new unique id and set $\rho_{j,n} = \rho_{i,n-1}$. If $c_{j,n}$ is an agent and its predecessors $c_{seed,d^*_{i,n-1}} \neq 0$, we set the individual world center of $c_{j,n}$ to $s_{j,n} = p_{j,n}$. This is useful to allow for unbounded worlds. If $c_{j,n}$ originated from a ψ symbol and any artifact $g_{h,n}$ has been placed by the same rule, we set $\rho_{j,n} = \iota_{h,n}$. This facilitates the tree-like structure of back-references. Concerning energy, all generated actors c_i are assigned a new energy value. Depending on the parameter $e_{suc,d^*_{j,n-1}}$ of their predecessor $d_{j,n-1}$, we provide four modes, how a successor's energy might be calculated: Constant energy, energy proportional to $e_{j,n-1}$, equal distribution of $e_{j,n-1}$ to successors, while loosing energy, and constant energy, if $e_{j,n-1}$ is not consumed by this, or else equal distribution. Notably, the latter two modes do not create but only redistribute or consume energy from the system. For the persisted predecessor, there are four analogue modes for $e_{persist,d^*_{j,n-1}}$: Constant energy loss, energy loss proportional to successor count, as well as the counterparts to the latter tow modes for e_{suc}.

Movement The second step of each iteration relies on Boid behaviour. All the following calculations are computed for every $d_{i,n'} \in \mathbb{D}_{n'}$. Should $\mathbb{D}_{n'}$ be empty, because the rewriting step has been skipped, we will operate on $\mathbb{D}_{n'} = \mathbb{D}_{n-1}$. Before calculating steering urges, we require the set of neighbours N_i of d_i. All actors within d_i's view cone, defined by d_{view,d_i^*} and $\beta_{d_i^*}$, similar to the one described in [9, Sec. 2.6.2], are considered neighbours. Through N_i, we can deduce the set S_i of neighbours deceeding the separation distance d_{sep,d_i^*}. We implemented the three basic boid urges as follows: Alignment aims to match d_i's neighbours' velocities in magnitude and direction; cohesion aims to reach the average position of actors in N_i; separation aims to move away from actors in S_i, weighting those closer more heavily. These urges all consider the influence $\eta_{c_j^*,d_i^*}$ of $c_j \in N_i$ in their averages. In contrast to the basic urges the following urges do not depend on the neighbourhood of d_i. The center urge $V_{cen,i} = s_i - p_i$ drives agents to their world center, keeping them interacting. Variety is introduced by $V_{rnd,i}$, a random but deterministic steering vector within the unit sphere. The constant bias vector $V_{bias,i} = \mathbf{1}$, together with its coefficient vector, describes a constant urge towards a certain direction. It may be used, e.g., to model gravity and thus gravitropism [3]—the phenomenon which results in many plant shoots growing upwards, overcoming gravity. Lastly, we introduced urges reacting to the local terrain. $V_{floor,i} = (0\ (p_{i,y} - h(p_i))^2\ 0)^\intercal$ describes a downward pressure, that increases with distance to the underlying terrain. Together with $V_{bias,i}$, this could be used to set a desired height, where both cancel each other out. Concerning the shape of the terrain, the following urges build primarily on the gradient. We compute them by use of an utility offset vector $o = (0.5 \cdot t_s\ 0\ 0.5 \cdot t_s)^\intercal$, which depends t_s, describing how far agents look, to determine a gradient. $V_{grad,i} = (h(p_i + o_x) - h(p_i - o_x)\ 0\ h(p_i + o_z) - h(p_i - o_z))^\intercal$ and $V_{slope,i} = (V_{grad,i} + V_{dws,i}) \cdot c$, where $c = -1$ if $|V_{dws,i}| < 0$ else $c = 1$ both describe the gradient of the terrain: $V_{grad,i}$ as a vector within the xz-plane, $V_{slope,i}$ adding a downhill component $V_{dws,i} = (0\ h(p_i - \frac{V_{grad,i} \cdot t_s}{|V_{grad,i}| \cdot 2}) - h(p_i + \frac{V_{grad,i} \cdot t_s}{|V_{grad,i}| \cdot 2})\ 0)^\intercal$. They allow agents to move towards valleys and mountain tops, either on the xz-plane or as if they walked or rolled. Lastly, $V_{norm,i} = (0\ h(p_i + o_z) - h(p_i - o_z)\ 1)^\intercal \times (1\ h(p_i + o_x) - h(p_i - o_x)\ 0)^\intercal$ represents the normal vector of the terrain and is, therefore, perpendicular to it. Through summation of all $V_{name,i}$, weighted by corresponding coefficients c_{name,d_i^*}, an acceleration vector is calculated. This vector is then multiplied per element with the constraint vector $\mathbf{c}_{const,d_i^*}$. Finally, the acceleration is truncated to a maximum amount of a_{max,d_i^*} and added to the agent's velocity, which, in turn, is clipped to V_{max,d_i^*}. Pace-keeping, a desire to move at v_{pace,d_i^*} controlled by c_{pace,d_i^*} and further described in [18], is applied to d_i. As the penultimate step in actual movement, the resulting velocity is stored in $v_{new,i,n'}$ and added to the current position, resulting in $p_{i,new,n'}$. If this position is below the terrain and $c_{noclip,d_i^*} = 0$, it is moved onto the terrain surface. To finalize the movement step we reduce each d_i's energy either by a constant amount or proportionally to the distance traveled, depending on the mode of e_{move,d_i^*}. After these calculations, we copy each $d_{i,n'} \in \mathbb{D}_{n'}$ as $d_{i,n}$ into our final result set $\mathbb{D}_n$, setting $p_{i,n} = p_{i,new,n'}$, $v_{i,n} = v_{i,new,n'}$ and $e_{i,n} = e_{i,new,n'}$.

Terrain Calculations This height of $h_n(p)$ is calculated by averaging the y-position of all actors c_i, weighted by their influence factor $\mu(c_i, p_j)$ on the terrain. This factor depends on the distance of c_i and p_j on the xz-plane and the influence $\eta_{c_i^*,terrain}$. We calculate it as $\mu(c_i, p_j) = (1 + \|p_j - p_i\|_{xz})^{-\eta_{c_i^*,terrain}}$. To disable influence of actor types, we set $\mu(c_i, p_j) = 0$, if $\eta_{c_i^*,terrain} = 0$. Notably, we employed a regularly spaced grid of sample points with spacing t_s for a more performant, cached, implementation. Therefore implementations might replace the height function in the movement step by the bilinear interpolation of the nearest sample-points. This representation also lends itself to visualization and export as a 3D-asset, due to easier triangulation.

4 Analysis of Local Model Behaviour and Dynamics

We investigated the space of local behaviour and dynamics of vSGs by sensitivity analyses of three minimal model configurations SG_1, SG_2, and SG_3 (Table 1).

4.1 Minimal Terrain Generation

SG_1 represents a minimal configuration capable of generating a basic terrain featuring plateaux, cliffs or mountains. We ran analyses on on a 300×300 area with $t_s = 2$. For SG_1, four agents of type Δ_1 occasionally placed Γ_1 artifacts at their position until iteration $n = 100$. Figure 2 shows the influence of $\eta_{\Gamma_1,terrain}$: Uniformly low values produce pointy features, values between 2 to 4 resulted in smooth features, and high values resulted in very distinct plateaus with cliffs between them, reminiscent of Voronoi diagrams.

The second property with large impact was the density of terrain-influencing actors. This has been discovered while examining sensitivity of the center urge coefficient. c_{cen,Δ_1} values above just 0.02 result in agents circling around their world center tightly, resulting in mostly flat terrains with high Γ_1-density hotspots, as seen in Figure 2. Such *detail clusters* could be used intentionally to mimic rocks. The observed behaviour with higher c_{cen,Δ_1} is due to the dominance of V_{cen} over other urges. Management of dominant urges was a recurring difficulty, especially in situations, where one urge was dominant but then is cancelled out by another.

4.2 Minimal Tree Generation

SG_2 entertains Δ_1 agents that place "wood" artifacts Γ_1 wherever they move, resulting in a 3D trace. Seen in Figure 3, the separation urge and distance d_{sep} control how new branches turn away from predecessors. The analysis shows that $|V_{sep}|$ is proportional to d_{sep} allowing for quick and slow separation behaviours with fixed c_{sep,Δ_1}. A normalized $\hat{V}_{sep}$ could simplify the interactions with the alignment urge, which counteracts the undirected growth by V_{sep}.

Low values of η_{Γ_1,Δ_1} result in an appearance of bushes, where higher values straightened branches by driving agents away. Cohesion particularly impacts the width of generated trees. Variations of both parameters can be seen in Figure 4.

Table 1. Parameter and production definitions for vSGs SG_1, SG_2, and SG_3. All parameters not mentioned are set to zero. $\mathbb{D}_0$ was given directly, for each.

SG_1	Terrain generation by agents Δ_1 through terrain forming artifact Γ_1					
c_{const,Δ_1}	$(1\ 0.5\ 1)^\intercal$	a_{max,Δ_1}	0.5	d_{sep,Δ_1}	5	$\Delta_1 \xrightarrow{11} \psi$
c_{cen,Δ_1}	0.001	v_{max,Δ_1}	2	$e_{persist,\Delta_1}$	$(const, 0)$	$\Delta_1 \xrightarrow{1} \psi\Gamma_1$
c_{rnd,Δ_1}	0.2	v_{norm,Δ_1}	1	e_{suc,Δ_1}	$(const, 10)$	
c_{pace,Δ_1}	0.1	d_{view,Δ_1}	100	e_{move,Δ_1}	$(const, 0.1)$	
c_{noclip,Δ_1}	1	β_{Δ_1}	180	e_{zero,Δ_1}	$(\Delta_1 \rightarrow \lambda)$	
$\eta_{\Gamma_1,terrain}$	8	t_s	2			
SG_2	Tree generation with tree agents Δ_1 and wood artifacts Γ_1					
c_{bias,Δ_1}	$(0\ 0.02\ 0)^\intercal$	c_{pace,Δ_1}	0.1	β_{Δ_1}	170	$\Delta_1 \xrightarrow{6} \psi\Gamma_1$
c_{const,Δ_1}	$(0.9\ 1\ 0.9)^\intercal$	η_{Δ_1,Δ_1}	1	d_{sep,Δ_1}	10	$\Delta_1 \xrightarrow{3} \psi\Gamma_1\Delta_1$
c_{sep,Δ_1}	0.9	η_{Γ_1,Δ_1}	0.1	$e_{persist,\Delta_1}$	$(const, 0)$	
c_{ali,Δ_1}	1	a_{max,Δ_1}	0.2	e_{suc,Δ_1}	$(inherit, 0.85)$	
c_{cen,Δ_1}	0.0001	v_{max,Δ_1}	0.6	e_{move,Δ_1}	$(dist, 0.2)$	
c_{rnd,Δ_1}	-0.002	v_{norm,Δ_1}	0.4	e_{zero,Δ_1}	$(10, \Delta_1 \rightarrow \Gamma_1)$	
c_{norm,Δ_1}	0.1	d_{view,Δ_1}	20	t_s	5	
SG_3	Water generation with droplets Δ_1 , clouds Δ_2, traces Γ_1 and water Γ_2					
c_{coh,Δ_1}	0.04	v_{max,Δ_1}	2.5	e_{suc,Δ_2}	$(inherit, 1)$	$\Delta_1 \xrightarrow{3} \psi$
c_{sep,Δ_1}	1.3	a_{max,Δ_1}	0.3	e_{zero,Δ_1}	$(0, \rightarrow \lambda)$	$\Delta_1 \xrightarrow{1} \psi\Gamma_1$
c_{ali,Δ_1}	0.5	β_{Δ_1}	120	e_{move,Δ_2}	$(const, 0)$	$\Delta_1 <_7 \Delta_1\Delta_1\Gamma_1 \xrightarrow{2} \psi\Gamma_2$
c_{rnd,Δ_1}	1.5	d_{view,Δ_1}	30	$e_{persist,\Delta_2}$	$(const, 0)$	
c_{floor,Δ_1}	0.1	d_{sep,Δ_1}	8	e_{suc,Δ_2}	$(const, 10)$	$\Delta_2 \xrightarrow{6} \psi$
c_{pace,Δ_1}	0.2	η_{Δ_1,Δ_1}	1	e_{zero,Δ_2}	$()$	$\Delta_2 \xrightarrow{1} \psi\Delta_1$
c_{slope,Δ_1}	0.3	e_{move,Δ_1}	$(const, 0.02)$	$\eta_{\Gamma_3,terrain}$	2	
v_{norm,Δ_1}	1.5	$e_{persist,\Delta_2}$	$(const, 0)$	t_s	10	
Parameter	Value	Parameter	Value	Parameter	Value	Productions

η_{Δ_1,Δ_1} and c_{rnd} were both required to be non-zero for interesting results, as some perturbation is required and $\eta_{\Delta_1,\Delta_1} = 0$ would mean Δ_1 agents do not see and therefore do not interact with each other. Investigating the impact of energy on tree growth, we varied energy consumption on movement and energy distribution on replication. The former controls overall tree size, while the latter only influences the size of offshoots (Figure 4). In order to evaluate the impact of terrain urges, we added two artifacts into our test scene, which generated a slope for the tree to grow on. Adjusting the four parameters $c_{floor} = 0.003, c_{norm} = 0.3, c_{bias,y} = 1.6$ and $d_{sep} = 6$, we generated sweeps of V_{floor}, V_{norm} and V_{bias} for this new $SG_{2'}$. The changes were required to establish a new reference tree, as the initial SG_2 did not yield insightful results in the revised environment. As shown in Figure 4, V_{floor} had a strong impact, especially as V_{norm} pushed agents towards the valley, indirectly increasing agents height. Besides this and

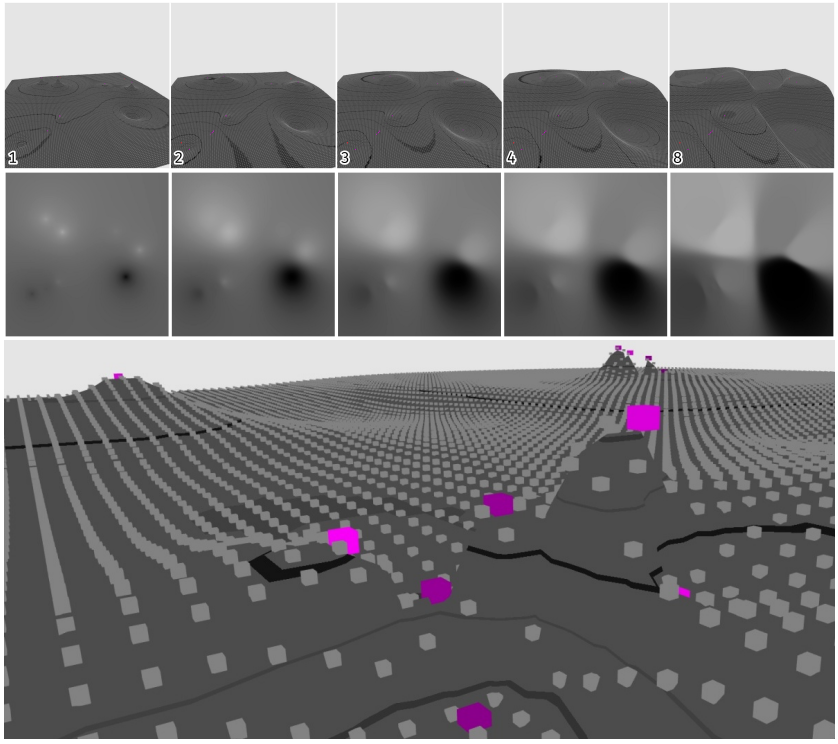

Fig. 2. Series of terrains and heightmaps of $SG_1(100)$, with varying $\eta_{\Gamma_1,terrain}$ (top and midde). Low values resulted in small peaks, while high values resulted in a clearly separated plateaus. The Γ_1 artifacts are shown in magenta. Screenshot of terrain details, generated by a high density of Γ_1 artifacts (magenta), placed by Δ_1 agents of SG_1 with $c_{center,\Delta_1} = 0.025$ (bottom).

an insensitive upwards push, we did not notice significant contribution of V_{bias} and V_{norm}.

4.3 Minimal Water Simulation

SG_3 features water droplet-like agents Δ_1 that leave wet spots Γ_1 and are stochastically dropped by cloud agents Δ_2. To visualize where water accumulated, we employed the production $\Delta_1 <_7 \Delta_1\Delta_1\Gamma_1 \xrightarrow{2} \psi\Gamma_2$. As the context distance is smaller than d_{sep}, Γ_2 are only placed, if agents collide forcefully. Particularly, this can happen by accumulating agents with $c_{slope} > 0$ in a basin. The sensitivity analysis showed that the physically inspired approach, i.e. following V_{slope}, works across a broad value spectrum as long as some perturbation is introduced.

Two intuitively insightful parameters were c_{coh} and c_{sep}, seen in Figure 5. Low cohesion and high separation resulted in agents not deceeding the context-

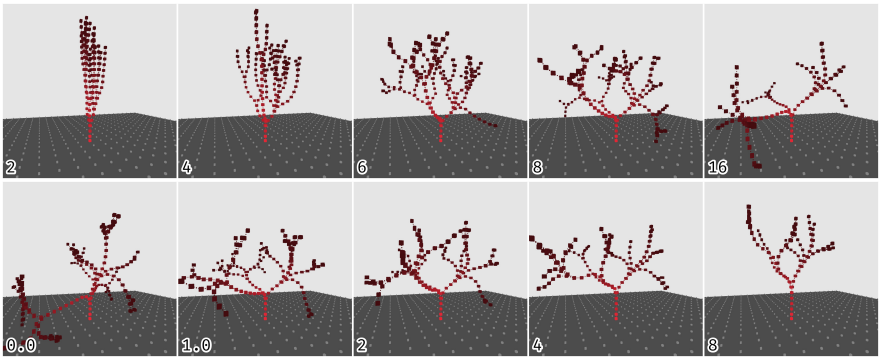

Fig. 3. Depictions of $SG_2(100)$ with varying values of d_{sep,Δ_1} (top) and c_{ali,Δ_1} (bottom). They showed a spectrum of orderly growth and uncontrolled sprawling with sharp turns, where V_{sep} dominated, highlighting that $|V_{sep}| \propto d_{sep}$.

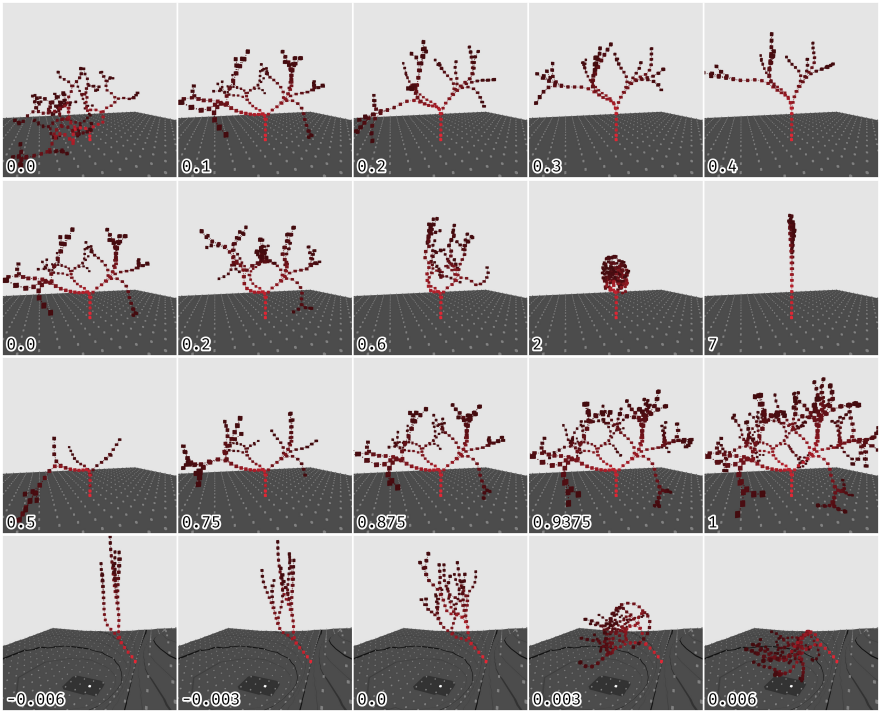

Fig. 4. Depictions of varying values of η_{Γ_1,Δ_1}, c_{coh,Δ_1}, and e_{suc,Δ_1} in $SG_2(100)$ and c_{floor,Δ_1} in $SG_{2'}(100)$ (top to bottom). Notably, c_{coh,Δ_1} controlled tree width, η_{Γ_1,Δ_1} straightened branches, and e_{suc,Δ_1} controlled offshoot length. Interactions between V_{floor}, V_{norm}, and V_{bias} can be observed in the last row as an oscillating motion, seen when $c_{floor,\Delta_1} > 0$.

distance, resulting in few Γ_2 artifacts. High cohesion resulted in groups of agents overcoming V_{slope}, wandering freely, and low separation resulted in very dense puddles.

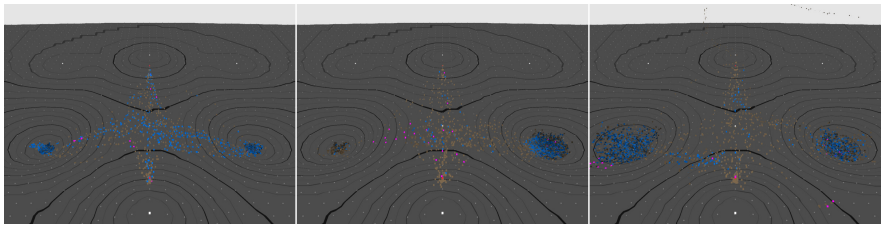

Fig. 5. $SG_3(1000)$ variations with $c_{sep,\Delta_1} = 0$ (left), $c_{sep,\Delta_1} = 0.5$ (middle), and $c_{coh,\Delta_1} = 0.4$ (right). Γ_1 are shown in brown, Δ_1 in magenta and Γ_2 in blue.

5 Global System Analysis

The analysis of minimal models highlights the inner workings of vSGs. However, an analysis with broader scope was needed to gauge the model's potential to generate large spaces. Hence, as a first step, we combined models from Section 4 into a coherent albeit basic virtual environment, discussing required modifications for successful integration. Next, we created a large space from the ground up, with the goal of higher visual interest and more interactions between species.

5.1 Combining Minimal Models

The goal of merging vSGs is to design a working, self-organising eco-system. The challenges involved are to link their respective agent types, to adjust influences and to distribute pre-existing agents. We conceived two ways to distribute tree agents and link them to water droplets: (1) Design sapling agents, with a rule context-sensitive to water or (2) presume a spread of seeds and let water spawn trees occasionally. Here, we present the latter approach, as the transport of saplings increases the simulation time. Either method generated trees laying flat, as seen in Figure 6. We ascribed this to inherited velocity and resolved the issue partially by introduction of an intermediary agent with $V_{max} = 0$ immediately replaced by tree agents. We called this intermediary agent a *biaser*, as it biases the velocity of a succeeding agent. Proximity to existing trees posed another problem as interferences of the separation urge between several trees impacted the growth processes (also seen in Figure 6). Hence, we introduced exclusivity, only allowing for tree growth with sufficient space. As there are no model mechanics for this yet, we regulated this by means of spawning frequencies encoded in the rules. Finally, as terrain can be reshaped any time, we encountered trees buried below terrain. Future revisions might implement *relative positioning*

to fix such structures, e.g., by maintaining height or the relative position to predecessors when the topography changes. For now, we coordinated the proper placement order through timed reproduction and energy depletion of the involved agents (Figure 6).

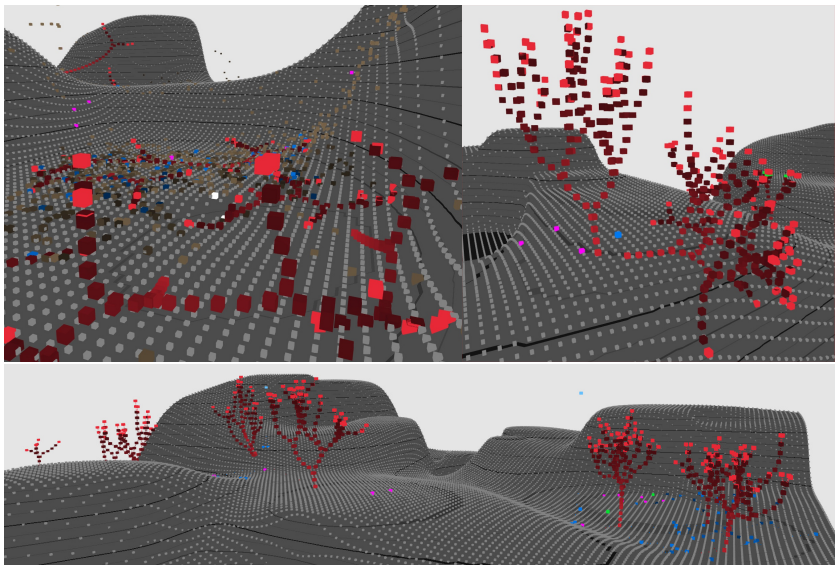

Fig. 6. Multiple trees (red) generated generated by infrequent tree placement through water droplets (magenta), without intermediary biasers (left), and without minimal tree generation distance (right). The final configuration (bottom) had a file size 2 times that of SG_3.

5.2 Creating a World from Ground Up

Aiming to generate a more detailed terrain, we created a regularly spaced grid of artifacts with $\eta_{terrain} \neq 0$, layered with other terrain shaping artifacts, placed by agents. Yielding results similar to those seen in Figure 7, instead of the aspired smoother natural topography, we removed the regular artifacts and added two more stages of agents, that are placed together with artifacts of the previous stage and in turn place other terrain shaping artifacts themselves. The results of stage 2 and 3 can be seen in Figure 8. As another layer of detail, we aimed to introduce grooves similar to hydraulic erosion. To disperse agents which might place such groves, we introduced hill-climbing agents with $c_{slope} < 0$. Reaching zero energy, they are replaced by an agent with $c_{slope} > 0$. Since terrain shaping artifacts placed by this second stage, would build walls instead of carving groves, we introduced a third stage with $c_{noclip} = 1$ and downwards bias, which finally placed artifacts. Results of both stage 2 and stage 3 agents are seen in Figure 8.

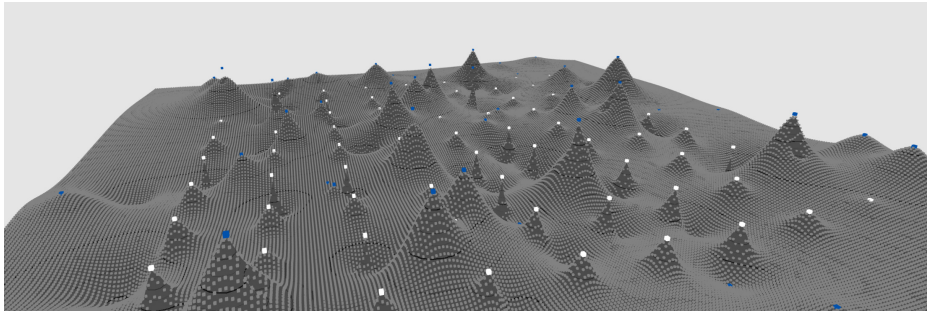

Fig. 7. Unexpected result of an attempt to create a smooth terrain through a grid of artifacts (white) and artifacts which were irregularly placed by agents (blue).

Proceeding with water generation, we again created a three stage process of cloud spawners, clouds and water droplets, each generating one after the other. To have the simulation terminate, we limited the agents' lifespans through energy consumption. To enhance simulation performance, we limited water artifact density by making use of our indirect implementation of exclusive context. The same technique has been used to limit water droplet generation rate, as they tended to stack below clouds, generating water artifacts mid-air. Concerning the flora, we took a hybrid approach of those mentioned in Section 5.1. By spawning saplings through droplet agents, we increased the probability of them being near water, while not requiring them to be attracted by droplets. If we were to choose a pure sapling approach, we would have had required an unsatisfactory tradeoff between saplings not reaching water and clusters of saplings, as separation and cohesion urges apply equally to both actors types. This could be addressed by introducing per-actor type urge coefficients, view and separation distances.

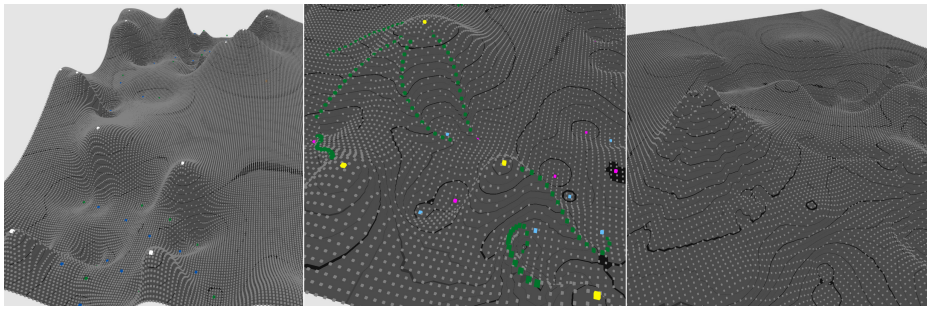

Fig. 8. Result of the three stage generation process without modification (left), unintended ridges (middle) and desired grooves (right).

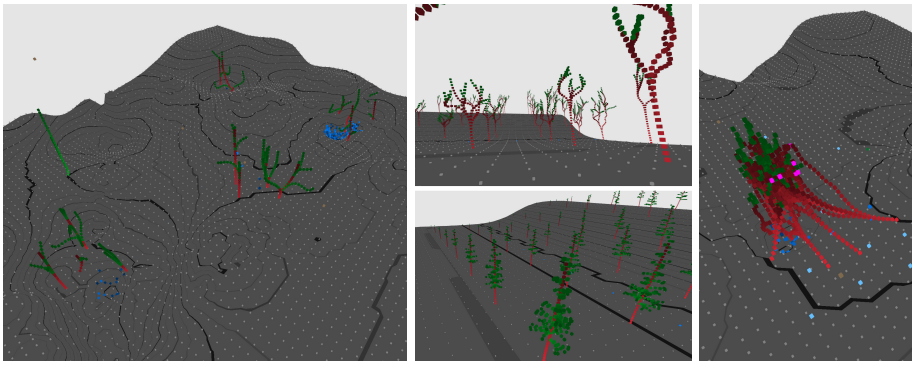

Fig. 9. A view into a more detailed vSG generated environment (left), views into a willow (top) and fir (bottom) farm, with changing terrain on one and varying starting energy on the other axis, and unexpected results of willow introduction into the more detailed environment (right).

The result so far is shown in Figure 9. We increased its appeal by introducing an agent to generate a "tree farm" (also seen in Figure 9). Here we could inspect the phenotypic plasticity of 100 identical genotypes exposed to variations in initial energy and the surrounding terrain. Generating firs through an upwards biased agent, which spawns other agents with heavily constrained vertical movement, was straight forward. A second pollard willow like tree, however, proved difficult. As seen in Figure 9, they tended to collapse when introduced into the scene, which emphasized the strong impact of the creation environment. As the farm environment ensured wide spaces between trees, cohesion had little effect, resulting in straight trees. A more severe problem arose, as we aimed to rework the willow into a weeping willow by means of a three stage agent production process (upwards, outwards, downwards). Consuming the initial energy to control the vertical size of the stem, we could not scale the width or length of branches. A solution would be to introduce intermediary agents, which would follow the artifact placing agents of each stage, without expending energy. Without them as context, they would be replaced by the agents of the next stage. This and solutions did not work robustly and cluttered up the initial database our environment stems from. A remedy could stem from further expansion of the underlying L-System towards a parametric L-System, where rules could read from and write to actor parameters. This extension generalizes energy, for example allowing custom depletion functions. Agent type A could, for example, implement a timer through $A \rightarrow A[t = t - 1]$, while death on zero energy could be implemented by a rule $A[e \leq 0] \rightarrow \lambda$. Together with proper disabling of rules, through a second, *negative* context, such an extension would increase implementation complexity, but would be in line with our core goal of world generation from a small initial database.

6 Conclusion

In this work we proposed a novel agent-based model for generation of virtual environments, featuring adaptive, unbounded world generation from a small database. We extended the existing SG model by increasing interaction with the environment, by allowing agents to form and react to the terrain, by abstracting generated data, and by diversifying the energy system. We analysed the proposed vSG model, through observation of changes in parameters in isolated agent types, gaining a basic understanding of possible behavioural impacts of changes. On this basis, we documented two of our attempts of creating a database for world generation, both featuring terrain, tree generation and water simulation. They showed satisfactory first results, as seen in Figure 10.

Throughout these analyses, we highlighted obstacles, e.g. flat trees and opaque configuration of water droplets, and resulting configuration patterns, e.g. timer and biaser intermediaries. As a result we identified negative context, relative positioning and parametric extension of the underlying L-System as promising additions. While further manual exploration of the creation spaces of this model would be beneficial, an evolutionary approach to exploring and optimising vSG configurations might circumvent usability issues. Besides, more elaborate, automated interpretation of generated point data would allow this model to advance from its experimental state, to be used in digital media.

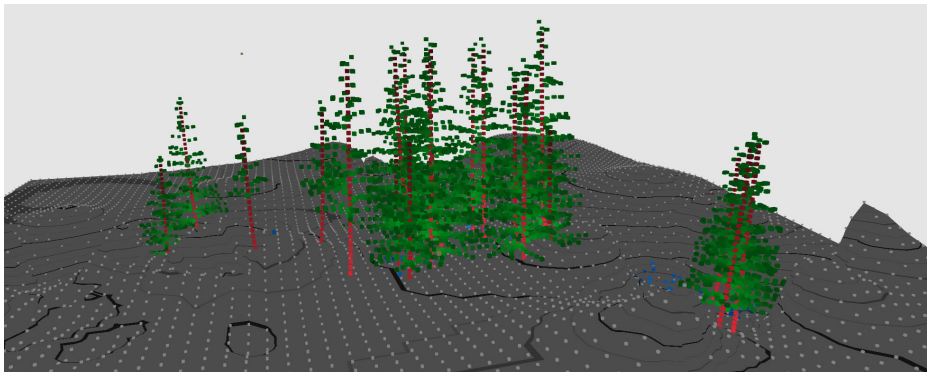

Fig. 10. A successful attempt to introduce firs into an existing environment. Configuration file size was 6 times that of SG_3, the most complex minimal model we used.

References

1. Bay 12 Games: Dwarf Fortress (2006), http://www.bay12games.com/dwarves/ accessed on 2020-11-19, current version 0.47
2. Costa, L.d.F., Rodrigues, F.A., Cristino, A.S.: Complex networks: the key to systems biology. Genetics and Molecular Biology **31**, 591 – 601 (2008). https://doi.org/10.1590/S1415-47572008000400001
3. Darwin, C.: The power of movement in plants. Appleton (1881)
4. Gearbox Software: Borderlands (2009)
5. Greuter, S., Parker, J., Stewart, N., Leach, G.: Undiscovered worlds–towards a framework for real-time procedural world generation **5**, 5 (2003)
6. Hello Games: No Mans Sky (2016)
7. Hendrikx, M., Meijer, S., Van Der Velden, J., Iosup, A.: Procedural content generation for games: A survey. ACM Trans. Multimedia Comput. Commun. Appl. **9**(1) (Feb 2013). https://doi.org/10.1145/2422956.2422957
8. Lindenmayer, A.: Mathematical models for cellular interactions in development i. filaments with one-sided inputs. Journal of Theoretical Biology **18**(3), 280 – 299 (1968). https://doi.org/10.1016/0022-5193(68)90079-9
9. von Mammen, S.: Swarm grammars: modeling computational development through highly dynamic complex processes. Ph.D. thesis (05 2009)
10. von Mammen, S., Jacob, C.: Genetic swarm grammar programming: Ecological breeding like a gardener. In: Srinivasan, D., Wang, L. (eds.) CEC 2007, IEEE Congress on Evolutionary Computation. pp. 851–858. IEEE Press, Singapore (2007)
11. von Mammen, S., Jacob, C.: The evolution of swarm grammars—growing trees, crafting art, and bottom-up design. Computational Intelligence Magazine, IEEE **4**, 10 – 19 (09 2009). https://doi.org/10.1109/MCI.2009.933096
12. Mojang Studios: Minecraft (2011)
13. Perlin, K.: An image synthesizer. In: Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques. p. 287–296. SIGGRAPH '85, Association for Computing Machinery, New York, NY, USA (1985). https://doi.org/10.1145/325334.325247
14. Prusinkiewicz, P.: Graphical applications of l-systems. In: Proceedings on Graphics Interface '86/Vision Interface '86. p. 247–253. Canadian Information Processing Society (1986)
15. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer-Verlag, Berlin, Heidelberg (1990)
16. Reynolds, C.: Flocks, herds, and schools: A distributed behavioral model. ACM SIGGRAPH Computer Graphics **21**, 25–34 (07 1987). https://doi.org/10.1145/280811.281008
17. Reynolds, C.: Boids — background and update (2001), http://www.red3d.com/cwr/boids/, accessed on 2020-11-19
18. Sayama, H.: Swarm chemistry. Artificial Life **15**(1), 105–114 (2009). https://doi.org/10.1162/artl.2009.15.1.15107
19. Schaal, J.: Procedural terrain generation. a case study from the game industry. In: Game Dynamics, pp. 133–150. Springer (2017)
20. Smelik, R.M., Tutenel, T., Bidarra, R., Benes, B.: A survey on procedural modelling for virtual worlds. Comput. Graph. Forum **33**(6), 31–50 (Sep 2014). https://doi.org/10.1111/cgf.12276
21. Worley, S.: A cellular texture basis function. SIGGRAPH '96, ACM, New York, NY, USA (1996). https://doi.org/10.1145/237170.237267